Calhoun: The NPS Institutional Archive
DSpace Repository

Faculty and Researchers

Faculty and Researchers' Publications

1995

Simulating belief systems of autonomous agents

Bhargava, Hemant K.; Branley, William C. Jr.

Decision Support Systems, Volume 14, 1995, pp. 329-348.

<https://hdl.handle.net/10945/43113>

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>

Simulating belief systems of autonomous agents

Hemant K. Bhargava^{a,1,*}, William C. Branley, Jr.^b

^a Code AS-BH Naval Postgraduate School 555 Dyer Road, Room 214 Monterey, CA 93943-5000, USA

^b U.S. Army A.I. Centre ATTN: SAIS-AI, 107 ARMY PENTAGON, Washington, DC 20310-0107, USA

Abstract

Autonomous agents in computer simulations do not have the usual mechanisms to acquire information as do their human counterparts. In many such simulations, it is not desirable that the agent have access to complete and correct information about its environment. We examine how imperfection in available information may be simulated in the case of autonomous agents. We determine probabilistically what the agent may detect, through hypothetical sensors, in a given situation. These detections are combined with the agent's knowledge base to infer observations and beliefs. Inherent in this task is a degree of uncertainty in choosing the most appropriate observation or belief. We describe and compare two approaches – a numerical approach and one based on defeasible logic – for simulating an appropriate belief in light of conflicting detection values at a given point in time. We discuss the application of this technique to autonomous forces in combat simulation systems.

Keywords: Belief simulation; Belief generation; Autonomous agent; Distributed interactive simulation; Belief revision; Defeasible reasoning

1. Introduction

Autonomous agents (or, *intelligent agents* Genesereth, Nilsson, 1987) are computer representations of real-world entities (e.g., humans) that can govern their own behaviour. They monitor their world, anticipate the consequences of their actions or of actions of other agents, and determine their own plan of action. Autonomous agents are beginning to appear in computer simu-

lation models, which are an important aspect of decision support technology. For example, imagine a simulation of a bank in which customers, instead of being points in a probability distribution, were computer programs capable of making their own decisions in the simulated world. For many such applications, we argue, the usefulness of the simulation as a decision support tool would be enhanced if the agents' belief processes were also simulated, i.e., they were based on realistic observation and detection capabilities which often may lead to *incomplete* and *imperfect* knowledge about the simulated world. In this paper, we present an approach to belief simulation that introduces, in a systematic way, imperfection in information made available to autonomous agents.

* Corresponding author.

¹ The work of this author was funded in part by LTC Herbert A. Dollahite (U.S. Army) of the Army A.I. Centre, and by Major David Neyland (U.S. Air Force) of DARPA/ASTO, with Michael J. Zyda (Naval Postgraduate School) as principal investigator. (File: belief-simulation-dss-5-94.)

Our work is significantly different from, but complements, current research on autonomous agents which has emphasized representation of, and reasoning with, an agent's beliefs but has largely ignored the question of how an agent develops these beliefs. There are several schemes for representing all sorts of meaningful information (including "common-sense knowledge") pertinent to the agent's world, and there are various techniques for reasoning with information –even when it may be imperfect or incomplete (see e.g., Pearl, 1988, Sanchez, Zadeh, 1987). Such representation and reasoning capabilities endow autonomous agents with "human-like intelligence." But, in simulation systems that are used as off-line training aids for real life decision making, it is equally important to get realism in the agent's observation and detection capabilities.

For example, in battlefield combat, decisions are often made because of imperfect information; e.g., in the extreme case, "friendly fire" can result from incorrect classification of an observed force or from an inaccurate belief of the location of a friendly force. In a computer simulation, an autonomous agent must receive its input through other computer programs or external sources; it has no sensors of its own to acquire information. If that is the case, and if it is desired that the agent *not* receive perfect knowledge, how exactly are the agent's beliefs to be determined? This is the question we seek to answer in this paper.

Our research is conducted in the context of autonomous agents in a combat simulation system. Specifically, we have implemented and tested our ideas on the NPSNET (Zyda, Pratt, 1991, Zyda et al., 1991) system. Military combat simulation systems are used to improve real-time decision making, train personnel, and analyze strategies, tactics and doctrines, through off-line simulations of battlefield combat. The rôle of autonomous forces (called "computer-generated forces") in these systems is to "populate" the world with an intelligent and ever-present opponent, creating a realistic and stressful environment for human players. Many such applications have been created, the most successful being ModSAF (Modular Semi-Automated Forces) and TacAir-SOAR (Tactical Air SOAR). See Cera-

nowicz, 1994a, Rosenbloom et al., 1993, Rosenbloom et al., 1994 for a discussion of these systems. The "simulator" –a component which acts as a monitor and referee –has complete access to facts about all the forces (human-controlled, or automated), battlefield terrain, communications, and other relevant factors. For effective training, it is not desirable that the simulator provide this same perfect information to the autonomous forces.

In the rest of the paper, we present our ideas in significant detail. In Framework, we propose a general framework for developing simulated beliefs for autonomous agents. We suggest conceptualizing the agent's activities in terms of three main steps: detection, measurement, and interpretation. The first two steps are concerned with acquiring and processing data in the environment, and are the focus of our paper. In our original implementation of this model, the third step, interpretation, was performed by the decision-making processes of the agent, which were separate from those processes that simulated agent beliefs. An overview of the agent's decision-making processes is given in Pratt et al., 1994. In Observation, we describe the features relevant to detection and measurement in the context of our application, that is, autonomous agents in combat simulation. We propose, in Belief-Generation, a general method for generating an agent's beliefs. We desire that the outcome, the beliefs, be influenced by evidence that is available and relevant to the task at hand. Therefore, all relevant pieces of evidence are considered and combined in making the final measurement. There are multiple ways to do so, and we discuss and compare two approaches –a numerical and a logical method –in Comparison. Finally, we conclude with some directions for future research.

2. Autonomous agents and belief systems framework

An autonomous agent's decisions and plans are influenced by the information it receives about the world in which it is operating. It is useful to classify this information into true *knowledge* about

the world and the *beliefs* that an intelligent agent forms about this world. A common distinction between knowledge and belief is that knowledge is simply that which is true in the world, while beliefs are an agent's understanding about the world Genesereth, Nilsson, 1987. Therefore, an agent can *believe* something that is false, but it cannot know that something is, for example, true, when in fact that thing is not true.

The distinction between knowledge and belief is important in modelling the dynamic aspects of the agent's operating environment. We can assume that the agent has a knowledge base Δ – a mostly static collection of facts acquired from various sources – but it must also continuously form beliefs $B(t)$ about the current state of its environment. In combat simulation, an autonomous agent's knowledge base is analogous to terrain maps, equipment manuals, and other such static information. It must form beliefs, however, about dynamic factors such as the enemy's position and intentions. Of course, beliefs can be represented in many ways, such as sentences in a logical language in much the same way as knowledge. Several “nonstandard” logics have been developed for representing and reasoning with both knowledge and beliefs. For the purposes of our paper, however, the choice of logic is unimportant. For simplicity, we assume a standard first-order logic language.

Formation of beliefs and making decisions based on reasoning with those beliefs are conceptually, and often physically as well, two separate functions. In a military environment, for example, observers form the “eyes and ears” of the commander, who must make decisions based on reports from many observers and many other non-human sources. Our aim here is to develop an appropriate method for *deriving* the input to the decision-making processes of an autonomous agent in a computer simulation. Since we are interested in simulating the agent's belief system, we characterize this process as *belief simulation*.

2.1. The role of belief simulation in agent reasoning

Our belief simulation model is “outside” of the agent in the sense that it is a force that acts

upon the agent by filtering its information. Our purpose in doing so is to answer the question “What should the agent believe in a given situation?” We argue that any reasoning entity is continuously performing three essential functions in an environment: *detection*, *measurement*, and *interpretation*. The detection activity involves the primitive senses, both natural and mechanical. It is concerned with the acquisition of stimuli, whether it be visual, aural, physical, etc. The measurement activity involves classifying, naming, identifying, locating and determining magnitude. This process answers such questions as “What is that sound?” or, “What is that object in the distance?” Measurement processes may use induction, deduction, or other forms of logic to accomplish their tasks. Finally, interpretation is forming conclusions and, possibly, making plans in response to the perceived object or its perceived behaviour.

During the detection, measurement and interpretation activities in both humans and autonomous agents, error can occur at many junctures. For example, detections may be faulty, either through sensing error or noise in the environment. Faulty detections may adversely affect both of the other processes. Measurement may be faulty due to incomplete knowledge, incorrect matching of stimuli with known patterns, or incorrect application of rules. Of course, measurement will also suffer when detections are faulty. Furthermore, faulty measurements may adversely affect interpretation. Interpretation error can also occur through either an incomplete rule base, an inability to assess the situation, an inability to choose the correct action or a combination of these and other factors. This is the most subjective of the three activities, and may be extremely sensitive to faulty information from the processes of detection and measurement.

Given the preceding discussion, we can now elaborate on a point made earlier about autonomous agents in a computer simulation. There is no requirement for such agents to perform detection and measurement processes. These agents can, if the application developer desires, perform their interpretation functions with the benefit of perfect knowledge about their environ-

Fig. 1. Summary of the belief simulation model. Note that the “indirect observations” $\Psi(t)$, are imbedded in the process labeled evidence combination rules.

ments. This is how many autonomous agent applications are designed.

The framework we have discussed allows us to place the belief simulation model, described in this paper, into perspective. It is a means of *simulating* the detection and measurement process of an autonomous agent. We call it a *belief simulator* because its output can be characterized as beliefs, as opposed to knowledge.

2.2. Belief generation: model overview

Our method for generating beliefs is depicted in Fig. 1 and summarized as follows. Let $\Theta(t)$ denote the true state of the agent’s environment at time t . Let Δ denote the agent’s knowledge base, and let $B(t)$ denote the agent’s beliefs about the environment at time t . Based on a) environmental conditions $\theta(t)$, and b) the agent’s apparatus δ for acquiring information, we first compute a probability distribution for the “detected value,” or “sensed value,” of each observable attribute α that is relevant at time t .

The agent’s detection –which represents the

output of hypothetical sensors possessed by the agent –is chosen by sampling this distribution. Let $\Phi(t)$ denote the collection of all of the agent’s detection values at time t . Then, for some suitably chosen function m , we can compute $\Phi(t)$ as

$$\Phi(t) = m(\Theta(t), \theta(t), \delta) \quad (1)$$

An agent may be able to directly detect certain attributes based on its sensors. For other attributes, it may only be possible to determine them indirectly based on knowledge about other attributes. Let $\Psi(t)$ represent the collection of these “indirect observations.” In our model, $\Psi(t)$ is determined via logical inference by combining the direct observations, $\Phi(t)$, with the agent’s knowledge base, Δ , as shown below.

$$\Phi(t) \cup \Delta \vdash \Psi(t) \quad (2)$$

Finally, due to this inferential process, an agent may perhaps have multiple observations (direct or indirect) relevant to a given attribute. If at a given time, an agent has only one observation and no prior beliefs relevant to an attribute, that observation becomes its belief at that time. Otherwise, if the agent has evidence favouring multiple values for an attribute observation (from multiple sources, multiple observations, or multiple inferences), the individual values are combined, using a set of rules γ , to get a final observation, $\Gamma(t)$, at that time.

$$\Phi(t) \cup \Psi(t) \vdash_{\gamma} \Gamma(t) \quad (3)$$

Finally, the agent’s belief system $B(t)$ at time t must be based on its observation at time t combined, using rules, τ with the agent’s prior beliefs. Due to the recursive nature of this definition, we can assume that all prior beliefs were summarized in the previous beliefs $B(t-1)$. Thus, the agent’s belief system at time t is obtained as indicated below.

$$\Gamma(t) \cup B(t-1) \vdash_{\tau} B(t) \quad (4)$$

There are, however, many ways in which multiple pieces of evidence leading to different conclusions may be combined. Each could potentially cause the agent to behave in a different manner. That is, we have many choices for the set γ . Similarly, there are many ways to formulate the

rules τ , candidates being Bayesian methods and weighted combinations of beliefs. A purpose in this paper is to describe and evaluate alternative formulations and applications of γ . Specifically, we compare a numerical approach, based on the Dempster-Shafer evidence combination rule (Gordon, Shortliffe, 1984), and a logical approach, based on defeasible reasoning (Nute, 1990, Nute, 1992). Other possibilities include path analysis Roehrig, 1995 and Bayesian methods Pearl, 1988. In the numerical approach, evidence values are combined mathematically in order to determine *cardinal* values of belief in each “plausible” conclusion. In the logical approach, a collection of defeasible rules represents the agent’s mechanisms for forming beliefs from observations, and meta-level strategies determine which rules best apply in the given circumstances. The calculus of defeasible reasoning is employed to determine the most plausible solution. We discuss the strengths and weaknesses of the two approaches and illustrate both with a simulation.

3. Modelling observation in autonomous agents

In this section we expand on the model presented in Overview. We start by covering preliminary information that must be presented in order to properly explain how the model is used to generate beliefs in Belief-Generation. To illustrate as we go along, we will use a simple scenario.

Battlefield scenario Our simulated battlefield has three kinds of moving objects: tanks, trucks and armoured personnel carriers (APCs). The vehicles are divided into two armies: the Argives and the Trojans.² Our autonomous agent, Ajax, is an Argive tank. Trojan vehicles are denoted by a *T*, Argives by an *A*. Tanks are denoted by a 1, APCs by a 2, and trucks by a 3. Thus, Ajax is an A1. Ajax is aware that tanks, trucks and APCs exist in his world, but

Table 1

Observable attributes and values. The values for *veh-name* are nomenclatures for specific models of combat vehicles. Thermal signature is the heat from a vehicle which reveals its general shape and other identifying characteristics

Attribute	Allowed values
location	(X,Y) co-ordinate values
speed	0...max speed of vehicle
veh-name	T1, T2, T3, A1, A2, A3
veh-type	tank, APC, truck
friend	yes, no
armament	main-gun, small-arms, none
has-armor	yes, no
silhouette	enemy-shape, friendly-shape
thermal-sig	enemy-sig, friendly-sig

does not know exactly when they will be present and where they are located. Ajax’s mission is to locate, correctly identify, and destroy all of the Trojan combat vehicles, i.e., Trojan tanks and APCs (T1s, and T2s). Due to ammunition constraints, Ajax *generally* avoids firing on trucks. For now, Ajax is the only autonomous agent on the simulated battlefield; all other vehicles are controlled by human operators of the simulation system.

3.1. A classification of attributes

The first step is to determine what attributes and events are of interest to the autonomous agent. In our scenario, the agent, or a human player, is able to directly observe or detect the vehicle characteristics listed in Table 1. In addition, the agent may be interested in certain other characteristics (e.g., amount of ammunition available to an enemy tank) that may be defined in the simulation system but which are not “observable” through any ordinary means.

For reasons that will become clear as we go along, we find it useful to classify these attributes according to the types of values they can have. Certain attributes are *continuous-valued* (e.g., *speed*) while others are *discrete-valued* (e.g., *has-armor*). This distinction influences how imperfection is introduced into the simulation of a belief. The fact that *has-armor* has only two choices, yes or no, means that the choice of which

² With due apologies to Homer Homer, 1969.

one to select as the simulated belief must be made much more carefully than when choosing a belief for *speed*, which has a range of floating-point values.

3.2. Knowledge, belief and performance

The next step is to identify the scope of the agent's world knowledge. For our simulated battlefield environment, permanent knowledge about the world is included in Ajax's static knowledge base, Δ . This is consistent with the knowledge possessed by a human soldier on a battlefield. In addition, a soldier would have common-sense knowledge about things on a battlefield. For example, tanks and APCs are made of armour, while trucks are not. He also knows that an enemy tank is more threatening than an enemy truck, because a tank possesses a main gun that fires a large calibre round over great distances. We grant Ajax the same degree of prior knowledge and commonsense knowledge as the typical combat soldier. In our application we represent this knowledge in first-order logic. For example,

- (1) $\forall x \text{ has-armour}(x) = \text{yes}$
 $\rightarrow \text{prob}(\text{veh-type}(x) = \{\text{tank}, \text{APC}\}, 1)$
 If x has armour, then x is either a tank or an APC.
- (2) $\forall x \text{ has-armour}(x) = \text{no}$
 $\rightarrow \text{prob}(\text{veh-type}(x) = \{\text{truck}\}, 1)$
 If x has no armour, then x is a truck.

Of course, the amount of commonsense and environmental knowledge that one chooses to represent is application specific. What is more relevant to this discussion is the agent's dynamic knowledge, or beliefs, about things in the environment that are subject to change, such as the exact location of a vehicle that is in motion.

One way of measuring the success of the belief simulation model is by studying the performance of the autonomous agent. As we will see, Ajax is less likely to be successful at locating, identifying, and destroying Trojan vehicles when he is acting upon imperfect information. We can judge the degree of imperfection by examining Ajax's beliefs at some time t during a sample simulation. In Table 2 we present simulation results showing that Ajax has incorrectly classified the vehicle

Table 2

A sample of Ajax's beliefs (vehicle 31, range = 4307 meters). The * denotes incorrect belief

Attribute	Actual value	Ajax's belief at time t
veh-type	tank	veh-type(31) = tank
* veh-name	A1	veh-name(31) = T1
* friend	yes	friend(31) = no
* location	(35843,25290)	location(31) = (36080,25390)
* speed	16	speed(31) = 13

under observation. At time t Ajax believes that Vehicle 31 is a Trojan tank, but at time $t + 1$ he may revise that belief based on new data. However, during the period from t to $t + 1$, Ajax may make a grave error (i.e., fratricide) based on his belief about the vehicle. This is how we achieve the effect of human error in our autonomous agent behaviour.

3.3. Factors affecting observation

Let us examine how battlefield conditions affect the capability to observe as well as the actual observation. On a modern battlefield, many things exist that affect the ability of people to detect and observe events in the environment. Noise, smoke, rain, terrain, fatigue, morale, and countless other factors can impair human performance. Our goal is to assess the overall impact of these factors so we can simulate their effect on autonomous agent performance. For this stage of model development, we have chosen five fundamental factors that, collectively, span a wide range of things that affect human performance. Other applications of the model might require many more factors, and different factors, than the ones we describe here. The factors for this application are distance, visibility, judgement, knowledge, and equipment. The first two represent the environmental conditions $\theta(t)$ at time t . The last three represent the agent's apparatus δ for acquiring and making inferences about information in the environment.

Distance: Denoted by d , this refers to the proximity of an event, object, or attribute of an object, in relation to the observer. The value of d ranges from 0 to 1: $d = 1$ when the event or object is so close that it can be positively identi-

fied by an astute observer, under good weather, and without the aid of any equipment; $d = 0$ when the object is so far away that there is no chance it can be detected or identified. The value is based on the actual distance d_a from the observer to the event or object, the distance $d(0)$ at which $d = 0$ for that particular event or object, and the distance $d(1)$ at which $d = 1$. These values are combined as follows: If $d_a \geq d(0)$ then $d = 0$; if $d_a \leq d(1)$ then $d = 1$; otherwise d is computed as shown below.

$$d = \begin{cases} \left(1 - \frac{d_a - d(1)}{d(0) - d(1)}\right) = \frac{d(0) - d_a}{d(0) - d(1)} & \text{if } d_a \in [d(1), d(0)] \\ 0 & \text{if } d_a \geq d(0) \\ 1 & \text{if } d_a \leq d(1) \end{cases} \quad (5)$$

The values for $d(0)$ and $d(1)$ are not the same for all attributes of a battlefield object. For example, an observer on a battlefield is more likely to detect the thermal signature of an approaching vehicle before judging the vehicle's speed and name. For each attribute, we determine a range $d(1)$ at which the attribute may be positively identified, and a range $d(0)$ at which each attribute is not even detectable. For development and testing of the observation model, we have compiled a table of $d(0)$ and $d(1)$ values for certain vehicle attributes (Table 3).³ Note that as we go down the list to attributes that are easier to observe, the observations can be made at greater distances.

Table 3
Distance (in meters) at which attributes begin to be observed ($d(0)$) and may be positively identified ($d(1)$)

Attribute	$d(0)$	$d(1)$
location	4000	500
speed	4000	500
veh-name	4000	800
veh-type	4000	1200
friend	4000	1200
armament	5000	1200
has-armor	6000	1200
silhouette	7000	1200
thermal-sig	8000	1200

Visibility: Denoted by v , its value falls between 0 and 1. This factor encompasses all things that impact on visibility: time of day, weather, smoke, dust, and haze. A value of 0 denotes the poorest possible conditions, such as dense fog or a moonless night. When $v = 1$, viewing conditions are perfect.

Judgement: A value between 0 and 1, j is an overall measure of the observer's physical and mental abilities. It refers to the observer's eyesight, hearing, alertness, intelligence, morale, and health. When $j = 1$, the observer is highly skilled and makes judgements that are extremely reliable. A value of 0 denotes the opposite.

Equipment: Denoted by q , this number is an overall measure of the utility of all equipment available to the observer. We refer specifically to equipment that aids in detection and observation, such as binoculars, night vision devices, and laser range finders. The naked eye is denoted by $q = 0$, while $q = 1$ means that the equipment affords maximum advantage to the observer.

Knowledge: This value, k , refers to certain categories of dynamic information that may be available to the observer. We said in K-B-performance that the autonomous agent possesses knowledge of many aspects of its environment. In addition to that knowledge, the agent may gain "privileged" information about its opponents or its environment. This corresponds to battlefield intelligence that is gathered and disseminated among military forces. The value, k , is a single measure of the value of all knowledge that can aid an observer or an autonomous agent. When $k = 1$, then the agent has perfect knowledge of that which is under observation or is subject to detection. In practice, when $k = 1$, then the agent "knows everything." When $k = 0$, then the agent has no relevant knowledge of the battlefield, other

³ These rough values are based partly on interviews with military personnel experienced in range estimation and combat vehicle identification. They are useful for testing this model. More specific data could be obtained through a rigorous data-collection effort.

than what is already permanently associated with the environment.

4. Generating an autonomous agent's beliefs

Having covered the key components of the model, we can continue with our scenario and explain how beliefs such as those in Table 2 are generated. We show how the battlefield factors are combined into a single probabilistic measure, m , of the ability of an agent to accurately detect a given stimulus under a given set of conditions. For some attributes, we can think of the chosen value as being sampled from a probability distribution –determined as a function of m –in which the correct value has a probability m . For other attributes, m is allowed to influence a reasoning process that ultimately results in belief selection for an attribute. This section is in the form of an overview. We present the functions that are used to compute m and then discuss how m is applied to the belief selection process for different classes of attributes. Finally, we explain how beliefs are updated, that is, how new observations are reconciled with previous beliefs.

4.1. Observation and belief functions

All five factors mentioned previously can be directly set, or influenced, by the autonomous agent programmer or user in order to force the agent to behave in a prescribed fashion. If these

factors are given low values, then the autonomous agent will be forced to work with incorrect or incomplete information most of the time. That is because all five of the factors combine to determine m , the overall ability to detect and manipulate stimuli. Conversely, if these factors are given high values, then the agent will have good information most of the time and will be very hard for a human to beat. The formulas that we have developed and used for computing $m(d, v, j, q, k)$ are given below.

$$c(d, v, j) = d^2 e^{-\lambda(1-j)v} (\lambda = 0.5) \quad (6)$$

$$g(c(d, v, j), q) = c + q(1 - c)^{2-q} \quad (7)$$

$$m(c(d, v, j), q, k) = (c + q(1 - c)^{2-q})^{1-k} = g^{(1-k)} \quad (8)$$

Combining these, we get

$$m(d, v, j, q, k) = (d^2 e^{-\lambda(1-j)v} + q(1 - d^2 e^{-\lambda(1-j)v})^{2-q})^{1-k}$$

Eq. 6 represents the effect of distance, visibility and judgement. All of them must be equal to 1 to get perfect ability to observe. As the observer's judgement deteriorates, the ability to observe decays exponentially. Eq. 7 captures the effect of equipment. It reduces to Eq. 6 when there is no useful equipment ($q = 0$), but otherwise an increase in q improves the ability to observe. Eq. 8 captures the impact of dynamic knowledge, which can lead to an m -value of 1 on its own. It reduces

Table 4
Ability to observe various attributes for object 31. ($k = 0.8$, $v = 1$, $j = 0.9$, $q = 0.5$)

Attribute	d	Actual	m	Complement	$1 - m$
location	0.0	(35843, 25290)	0.81	n/a	0.19
speed	0.0	16 kph	0.81	n/a	0.19
veh-name	0.0	A1	0.81	{A2, A3, T1, T2, T3}	0.19
veh-type	0.0	tank	0.81	{truck, APC}	0.19
friend	0.0	yes	0.81	{no}	0.19
armament	0.18	main-gun	0.82	{small-arms, none}	0.18
has-armor	0.35	yes	0.84	{no}	0.16
silhouette	0.46	friendly-shape	0.86	{enemy-shape}	0.14
thermal-sig	0.58	friendly-sig	0.89	{enemy-sig}	0.11

to Eq. 7 when there is no dynamic knowledge, and an increase in k has a positive impact on m .

All of this is consistent with our requirements and definitions of the factors. It can be verified easily, by comparing first-and second-order partial derivatives, that the m function has the desired behaviour. In particular, it may be seen that the function behaves appropriately at each end point (i.e., by setting each factor to 0 and to 1), and that the value of m changes in the correct direction throughout the interval $[0,1]$ as we vary each factor (by computing the first and second derivatives). We have also experimentally confirmed that these equations exhibit desired behaviour. However, we do not argue that these equations are the only or optimal ones for this application; rather we claim only that they are sufficiently useful for our purpose. We illustrate the use of these functions in the following example.

Example 1. Determining ability to observe. Consider an Argive tank located 4037 meters away from Ajax at time t . This constitutes the true state, $\Theta(t)$. The m values in Table 4 reflect Ajax's ability to detect and manipulate stimuli in the form of object attributes. The value $1 - m$ is assigned to the complement of the actual value of each attribute.

Ajax's detections $\Phi(t)$ are obtained using the m value for each attribute. Recall from At-

tribute-Classification that the attributes are classified according to the types of values they can have (i.e., continuous or discrete). Fig. 2 depicts how the m -value changes as we vary distance, over two different settings for k . In the rest of this section, we explain how m is used to select beliefs for each class of attributes.

4.2. Introducing error into observations: A method for non-discrete cases

We first examine how m is used to select observations and beliefs about a continuous-valued attribute, *location*. Observing location involves determining a distance and direction from the observer (i.e., the agent) to the target. An error may occur in determining one or both of these. To model this, we will choose the perceived location from a set of points contained in the area of a circle, with the true value as its centre, and a radius that is a function of the m value. The distance error is computed as a percentage of the true distance between the agent and the target. For example, if the target is 4,037 meters away, and $1 - m = 0.19$, then the diameter is $\epsilon = 0.19(1000)$ or 190 meters. Thus, the maximum range error is $\epsilon/2 = 95$ meters.

Next, we select a number at random from the interval $[0, \epsilon/2]$. This number is the distance error we introduce into the location belief. We

Fig. 2. m values and error in location.

apply the distance error in a random *direction* from the true target location. In this example, if the number chosen from the interval is 50, then we compute (X,Y) coordinates for some point that is 50 meters in a random direction from the target. That gives us Ajax's observation about location of vehicle 31 at time t . If Ajax has no other information relevant to vehicle 31's location, his belief at time t is identical to this observation. However, when Ajax makes the next observation at time $t + 1$, his previous belief must be considered. We accomplish this by setting

$$Bel_{\alpha}(t + 1) = \rho \cdot \phi_{\alpha}(t + 1) + (1 - \rho) \cdot Bel_{\alpha}(t) \quad (9)$$

where Bel indicates belief, ϕ is the observation, and ρ is a number between 0 and 1 indicating the distribution of weight between current observation and prior belief. This technique allows us to introduce imperfection in beliefs while maintaining a certain degree of consistency across multiple observations.

The effect of our scheme for computing beliefs for location can be seen in Fig. 2. This graph depicts data that was gathered from two separate runs of the location belief model. In one case, the knowledge parameter, k , was set to 0.8, while in the other $k = 0.3$. For both sets of data, the graph displays the amount of error, in percentage terms, in relation to the distance from the agent to the target. Also shown is the general trend of m values in relation to distance. First notice that when the distance is great, m is relatively low. When m is low, there is a greater chance of error. This can be seen in the general downward trend of the data points from left to right, which corresponds to low m values vs. high values. Secondly, notice the impact of knowledge on error. When $k = 0.8$, the graph is noticeably less erratic than when $k = 0.3$. Intuitively, this corresponds to the notion that a more knowledgeable observer, one with better information, will be somewhat more consistent over the long term.

4.3. Combining multiple pieces of evidence

Next we examine belief generation for attributes whose values may be inferred from val-

```

Goal: veh-name is in {T1,T2}
Conditions:
1. veh-type is in {tank, APC}
Subconditions:
  a. armament is in {main-gun, small-arms}
  b. has-armor = yes
2. friend = no
Subconditions:
  a. silhouette = enemy-shape
  b. thermal-sig = enemy-sig

```

Fig. 3. Decision-making criteria for Ajax

ues of other attributes. These are the discrete-valued attributes covered in Attribute-Classification. We will employ a reasoning system to infer values of these attributes since the enumerated values of these attributes are not suitable for the kind of approach discussed in the previous section. For instance, if an attribute is associated with a set of three allowable values, how does one directly introduce a five percent error? Furthermore, a logical approach exploits ways in which these attributes are related to each other.

Recall that Ajax's mission is to identify and destroy all Trojan combat vehicles. We say that in order to identify a vehicle, its attribute *veh-name* must be determined. Furthermore, since Ajax only wants to destroy combat vehicles, he must determine that the vehicle name is in the set {T1,T2}. The conditions and preconditions necessary for Ajax to do this are summarized in Fig. 3. Note that these conditions and preconditions refer to other discrete-valued attributes. In the reasoning methods that we introduce in this section and expand upon in Comparison we use these relationships to arrive at "logically correct" belief values for the discrete-valued attributes.

4.4. Resolving new beliefs with past beliefs

As in the non-discrete case, once tentative beliefs and conclusions have been determined using the methods above, we must consider the agent's previous beliefs before making a final selection. The simplest case is the first observation at time t_0 , where there are no prior beliefs. In this case, the agent's belief is the same as the

current tentative observations or conclusions. In the next case, prior beliefs exist but they do not contradict the latest observation. In this case the equation for the simplest case holds. In the general case, however, the observations $I(t+1)$ are contradicted by $B(t)$. The ways in which we handle this are different for the two approaches. They will be illustrated in the following section.

5. A comparison of numerical and logical approaches to nonmonotonic reasoning in belief simulation

To illustrate a logical approach and a numerical approach for choosing appropriate beliefs for an agent, we will continue with our example and show how a vehicle identification belief would be chosen under each method. We will calculate beliefs for a single point in time at a distance of 4037 meters. The values in Table 4 will be used for both examples. Following the step-by-step illustration, we will discuss the results of several simulations that were conducted for the purpose of assessing the utility of each approach. We will explain what we feel are the desired characteristics in a model such as this, and then show, using data from the simulations, how we evaluated the two approaches against these characteristics.

Recall that a vehicle identification consists of the attribute *veh-name*, which is determined by *veh-type* and *friend*. These, in turn, are determined by other attributes, as laid out in Fig. 3. We will follow this strategy in our examples.

5.1. Employing a numerical technique for choosing beliefs

In our first approach, the numerical one, we employ the Dempster-Shafer rule of evidence combination Shafer, 1976, Gordon, Shortliffe, 1984. This involves computing a numerical value to determine the degree of support, or evidence, provided by one attribute value for another, and then combining and normalizing these values in order to arrive at a figure that captures the net impact of all relevant evidence. Furthermore, we distinguish between two kinds of evidence: that provided by *direct observation* ($\Phi(t)$ of Framework, Eq. 1) –when a sensory device is used to detect the attribute –and that provided by *indirection observation* ($\Psi(t)$, Eq. 2), which applies to information gained through a combination of inference and detection. The agent's knowledge base Δ provides the necessary rules for computing evidence values for indirect observations using m (from Table 4), and Eqs. 2 and 3 presented in Framework. These computations, along with the remaining steps of this procedure, are explained below.

- (1) Determine evidence, $\phi_{\text{veh-type}}(t)$ and $\phi_{\text{friend}}(t)$, from direct observation of *veh-type* and *friend*, respectively. We have m -values of 0.81 for both attributes. We interpret them as partial evidence (0.81) for the correct values of *tank* and *yes*, and partial evidence (0.19) for their complements. A sample of this is shown as m_1 in Table 5. This table illustrates computations only for *veh-type*; however,

Table 5
Sources of evidence for the vehicle type of Object 31

Evidence	Source
$m_1(\text{veh-type}(31) = \{\text{tank}\}) = 0.81$	direct observation (m)
$m_1(\text{veh-type}(31) = \{\text{truck, APC}\}) = 0.19$	direct observation ($1 - m$)
$m_2(\text{veh-type}(31) = \{\text{tank, APC}\}) = 0.84$	Rule 1 (has-armor)
$m_2(\text{veh-type}(31) = \{\text{truck}\}) = 0.16$	Rule 2 (has-armor)
$m_3(\text{veh-type}(31) = \{\text{tank}\}) = 0.82$	Rule 3 (armament)
$m_3(\text{veh-type}(31) = \{\text{truck, APC}\}) = 0.18$	Rule 4 (armament)

Table 6

Combining m_1 and m_2 ($m_1 \oplus m_2$)

	$m_2\{\text{tank,APC}\} = 0.84$	$m_2\{\text{truck}\} = 0.16$	$m_1 \cdot m_2(\{\}) = 0.13 \quad \kappa = 1 - m_1 \cdot m_2(\{\}) = 0.87$
$m_1\{\text{tank}\} 0.81$	$\{\text{tank}\} 0.68$	$\{\}$ 0.13	$m_1 \oplus m_2(\{\text{tank}\}) = (0.68/\kappa) = 0.78$ $m_1 \oplus m_2(\{\text{truck}\}) = (0.03/\kappa) = 0.04$
$m_1\{\text{truck,APC}\} 0.19$	$\{\text{APC}\} 0.16$	$\{\text{truck}\} 0.03$	$m_1 \oplus m_2(\{\text{APC}\}) = (0.16/\kappa) = 0.18$

those for *friend* would be performed in the same manner. The general rule for doing this computation is simple. For any attribute α and object x , if the true value of $\alpha(x)$ is η and if the m value for α is p , then there is evidence p indicating that $\alpha(x)$ equals η , and evidence $1 - p$ indicating the complement of $\{\eta\}$.

- (2) Determine evidence $\psi_{\text{veh-type}}(t)$ and $\psi_{\text{friend}}(t)$ for the two attributes that can be inferred using Ajax's knowledge base Δ and other observations in $\Phi(t)$. For example, consider the effect of an observation about armour: we know ($m(\text{has-armour}(31)) = 0.84$). We also know (from Rules 1 and 2) that if a vehicle x has armour, then it must be a tank or an APC (probability 1), and that if x does not have armour it must be a truck. Thus, we can logically infer additional evidence m_2 about vehicle type: 0.84 (multiplying 0.84 and 1) for tank or APC, and 0.16 for truck. Similarly, the values for m_3 are obtained using the "armament" rule. These inferences about the value of vehicle type constitute $\psi_{\text{veh-type}}(t)$ in this example. The same procedure would apply to *friend* using evidence from the "silhouette" and "thermal-sig" rules. In general, these inferences are made (recall Eq. 2) by combining the agent's knowledge base with

current direct observations, and using the following additional rule of inference. Suppose that we have $m(\alpha(x)) = p$, and that the agent's knowledge base contains a statement of the form $(\alpha(x) = \eta_i) \rightarrow (\text{prob}(\phi, q_i))$ for some value (set of values) η_i . From this we first compute, as in step 1 above, the evidence p_i for the value η_i . Then, combining this value with the antecedent of the statement, we can infer evidence $p_i \cdot q$ for the statement ϕ . We compute all such conclusions that logically follow from $\alpha(x)$.

- (3) Combine the individual evidences. In general, independent observations and inferences could lead to multiple, possibly contradictory, values for an attribute. For example, according to our knowledge base, "has-armour(x) = no" and "armament(x) = main-gun" are contradictory, since the presence of a main gun indicates a tank, while the absence of armour indicates a truck. Therefore, a method for reasoning about multiple, and possibly contradictory, pieces of evidence is required. Recall that such a method constitutes our set of rules γ (see Eq. 3). Under our numerical approach, we have the Dempster-Shafer method Gordon, Shortliffe, 1984, Shafer, 1976 for this purpose. This results in the following computations.

Table 7

Combining m_{12} and m_3 ($m_{12} \oplus m_3$)

	$m_3\{\text{tank}\} = 0.82$	$m_3\{\text{truck,APC}\} = 0.18$	$m_{12} \cdot m_3(\{\}) = 0.32 \quad \kappa = 1 - m_{12} \cdot m_3(\{\}) = 0.68$
$m_{12}\{\text{tank}\} 0.78$	$\{\text{tank}\} 0.68$	$\{\}$ 0.14	$m_{12} \oplus m_3(\{\text{tank}\}) = (0.64/\kappa) = 0.94$
$m_{12}\{\text{APC}\} 0.18$	$\{\}$ 0.15	$\{\text{APC}\} 0.03$	$m_{12} \oplus m_3(\{\text{APC}\}) = (0.03/\kappa) = 0.04$
$m_{12}\{\text{truck}\} 0.04$	$\{\}$ 0.03	$\{\text{truck}\} 0.01$	$m_{12} \oplus m_3(\{\text{truck}\}) = (0.01/\kappa) = 0.02$

- (a) Combine m_1 with m_2 as shown for attribute *veh-type* in the tableau in Table 6. Notice that, as we “add” the effect of m_2 to m_1 , the probability of correctly observing *veh-type* decreases from 0.81 to 0.78. Now we go on to add the effect of the armament rule. For conciseness of notation let us denote $m_1 \oplus m_2$ by m_{12} .
- (b) Combine $m_1 \oplus m_2$ with m_3 as shown in Table 7. Now the probability of observing a tank is 0.94, considerably higher than it was without considering additional evidence.
- (4) Make random selection and assert observation. We have three possibilities for *veh-type*: tank (0.94), APC (0.04) and truck (0.02). We choose a random number y from a uniform distribution (1 to 100), and determine Ajax’s observation as follows:
 - (a) If $1 \leq y \leq 94$, then $\text{veh-type}(31) = \text{tank}$
 - (b) If $94 < y \leq 98$, then $\text{veh-type}(31) = \text{APC}$
 - (c) If $98 < y \leq 100$, then $\text{veh-type}(31) = \text{truck}$
 In this example, the first statement above was satisfied and the corresponding observation was asserted. If a number between 95 and 100 had been generated, Ajax would have ended up with an incorrect observation about vehicle type. The same procedure was applied to attribute *friend* with the results: yes (0.06) and no (0.96). When we made a random selection we chose “no.” With these two pieces of information we can now deduce a value for the attribute *veh-name*. Since “tank” is denoted by a 1, and the enemy is denoted by “T,” the corresponding vehicle name is T1.
- (5) Compare current observation with previous belief. It so happens that this observation does not agree with our belief at time $t - 1$. We make a choice by once again combining evidence values in an intersection tableau, using current observations and previous beliefs as sources of evidence. It is only necessary to reconcile those attributes that are different. In this example, the attributes *armament*, *veh-type* and *veh-name* were different at time $t - 1$, due to the effects of random selection. At $t - 1$, *armament* = small-

arms, *veh-type* = APC, and *veh-name* = T2. When we recombine these in a tableau, we use as supporting values all *current* observations that pertain to these attributes. For example, we will gather current evidence supporting *armament* = small-arms, and current evidence supporting *armament* = main-gun. In this example, we chose *veh-type* = tank, *armament* = main-gun and *veh-name* = T2 as final values.

- (6) Transmit final belief to the agent’s “decision maker.” The actual work of the belief simulation model is complete at the previous step. However, we mention this final step as a reminder of the purpose of this effort. We want to communicate our results to the agent’s interpretation processes (see Framework). In this case, due to the effects of random selection, the value T2 for vehicle name is not consistent with the other attribute values that describe the object, since T2 denotes a Trojan APC while *armament* = main-gun implies a tank. We impose no requirement for consistency among the attribute values of an object. It is up to the interpretative processes to make the best of this “faulty” input data. However, we also recognize that this may vary from one implementation to another. For example, another approach in this final step would be to assert an incorrect value for the attribute *veh-name*, which would have the effect of incorrectly identifying the vehicle, but then adjust the remaining discrete-valued attributes so that they are consistent with *veh-name*. This would allow the agent’s decision-making and planning processes to be greatly simplified.

In this example we have demonstrated a numerical approach to choosing beliefs for an agent. We calculated current observations based on evidence combination and then combined observations with previous beliefs to determine our final beliefs. We then considered some issues pertaining to the actual contents of the belief message that is transmitted.

We showed how evidence combination was used to determine observations about *veh-type* and *friend*. In our implementation, we adopt a

strategy that minimizes computation time by calculating only those attribute observations that are necessary to identify the vehicle (i.e., determine its *veh-name*). However, other implementations may perform the evidence combination steps for *all* attributes in order to obtain the most thorough overall observation and subsequent belief. This is a valid use of the model. The approach taken should be governed by the purpose of the application.

5.2. Employing a defeasible reasoning technique for choosing beliefs

Now we will step through exactly the same vehicle identification problem that we used for the numerical approach, only this time we will attempt to logically infer what the belief values should be. Although we are calling this the “logical” approach, we are not abandoning random selection altogether. Recall our overall framework from Framework: detection, measurement and interpretation. We said that the belief simulation model simulates detection and measurement functions for the agent. We also pointed out how errors can occur in the detection process though faulty sensing or noise in the environment, and that those errors may adversely affect measurement and interpretation. In the logical approach to choosing beliefs, we employ random selection at the detection level to simulate the overall effect of both faulty sensing and noise. We then make logical inferences using these potentially faulty detection values.

In our simulation of the agent’s measurement processes, we try to make the best inference that we can, given the detection values that are received. Following the technique of defeasible reasoning Nute, 1992, the rule base we use for this purpose contains a mixture of three kinds of rules: *absolute rules*, denoted by $\rightarrow$; *defeasible rules*, denoted by $\Rightarrow$; and *defeaters*, denoted by $\rightarrow$. Traditional rule bases contain only absolute rules that are consistent with each other and with factual statements in the knowledge base. New assertions that cannot be logically supported are rejected. Defeasible rules, on the other hand, have antecedents that can be defeated in special

cases. The special cases are called defeaters. The application of a defeasible rule may also be blocked by conflicting defeasible rules. This kind of reasoning system permits one to reason with such notions as “maybe,” “usually,” “typically,” (in contrast to the universal quantification in standard logic) and so on, and is a form of nonmonotonic reasoning. There are other approaches to this style of reasoning that are summarized succinctly in Brewka, 1991.

The rule base that we will refer to in this section is essentially a restatement of the agent’s primary knowledge base Δ . We have dropped the measures of probability in the consequents of the rules and restated many of them as defeasible rules. For example, the original rule

$$(\forall x \text{ has} - \text{armour}(x) = \text{no}) \\ \rightarrow (\text{prob}(\text{veh} - \text{type}(x) = \{\text{truck}\}, 1))$$

is replaced by the defeasible rule (since there might be vehicles, other than trucks, without armour)

$$(\forall x \neg \text{has} - \text{armour}(x)) \Rightarrow (\text{truck}(x))$$

and

$$(\forall x \text{ armament}(x) = \text{small} - \text{arms}) \\ \rightarrow (\text{prob}(\text{veh} - \text{type}(x) = \{\text{tank}, \text{APC}\}, 1))$$

is replaced by the defeasible rule

$$(\forall x \text{small} - \text{arms}(x)) \Rightarrow (\text{tank}(x) \vee \text{APC}(x))$$

Since the rule base now contains a mixture of absolute and defeasible rules, the logical paths of inference can be said to have “strict” and “non-strict” links Nute, 1992. The links connect nodes, and each node represents an attribute value, such as (veh-type = tank). Also note that the antecedents of the rules are themselves attributes of the object under consideration and thus have *m* values associated with them. These characteristics of our rule base are used to form meta-level strategies for reconciling conflicting detection values.

- (1) The Strictness Rule. If two or more paths compete, and one path has the fewest non-strict links, then that path is superior.
- (2) The Reliability Rule. If two or more competing paths have the same number of strict and

non-strict links, and one path has a node with the highest m value, then that path is superior. This rule is based on our interpretation of m , namely, as a measure of the likelihood of correctly identifying an attribute.

- (3) The Persistence Rule. If two or more competing paths have the same number of strict and non-strict links, and no path has a node with the highest m value, then a decision may be postponed until an attribute value has occurred n times in n decision cycles. When that occurs, the path containing that value is the superior path. This is analogous to a “wait and see” approach. The agent’s interpretation processes will receive no information until the conflict is resolved. The amount of time that this can continue is influenced by n , which is user-settable. It is also influenced by the random selection of detection values at the start of each decision cycle. This strategy may be abandoned if, while waiting for the necessary conditions to be satisfied, another strategy can be applied successfully.

One result of applying these strategies is that the reasoning system is now equipped to handle cases for which no prior knowledge or rules existed. For example, suppose the agent encounters a truck that is armed with machine-guns. In the original rule base this assertion would be rejected because only tanks and APCs have weapons. It could also be rejected because a truck is not an armoured vehicle, while tanks and APCs (which carry weapons) have armour. In the defeasible rule base, however, if the m value for armament = machine-gun is equal to the m value for has-armour = none, and the paths containing those nodes have equal numbers of strict and non-strict links, then Rule 3, above, will apply and the model will take a “wait and see” approach. If those same attribute values occur n times, then the model will modify its world knowledge base to include the fact that trucks may be armed (and presumably dangerous).

Now that we have given an overview of our defeasible logic approach, we can proceed with the vehicle identification example outlined at the beginning of this section.

- (1) Calculate m for the attributes *armament*,

has-armour, *silhouette*, and *thermal-sig*. Select a detection value for each attribute from the respective sets of possible values (i.e., {main-gun, small-arms, none} for *armament*). Allow a weight of m for the true value, and $1 - m$ distributed evenly among the remaining values. (Note: This is implementation-specific; $1 - m$ may be distributed according to other rules.)

- (2) Assert the resulting attribute detections. For this example we assert

silhouette = friendly-shape ($m = 0.86$)

thermal-sig = enemy-sig ($m = 0.89$)

armament = main = gun ($m = 0.82$)

has-armour = yes ($m = 0.82$)

- (3) Deduce values for the attributes *veh-type*, *friend*, and *veh-name*. The last of the three is, of course, what we wish to determine. Note that we have a problem inferring whether the vehicle is friend or foe, based on the information provided. When we apply the first meta-rule, we discover that the paths containing these nodes have the same numbers of strict and non-strict links, so that rule doesn’t resolve the conflict. However, the attribute *thermal-sig* has a higher m value, which satisfied Rule 2 and allows us to assert friend = no. Finally we can assert *veh-name* = T1. The resulting set of attribute values becomes our current observation.

- (4) Reconcile current observations with previous beliefs. We compare the attribute observations at time t with beliefs at time $t - 1$ to determine which attribute values, if any, are different. In this example, we found that at time $t - 1$, friend = yes and *veh-name* = A1. This means that the vehicle we previously believed to be a friendly tank is now appearing to be an enemy tank. To resolve this we employ a backtracking procedure to determine whether or not the previous beliefs can be logically supported under the current conditions. If necessary, the backtracking procedure may invoke the meta-rules to resolve an impasse. At some point, the previous belief will be accepted or rejected. If it is rejected, then the choice is easy: we assert the current observation as the current belief and move to

the next step. (Even though several decision cycles may have passed if, during the backtracking phase, Rule 3 was invoked and the model went into its “wait and see” mode.) The situation is more complicated in the case when a previous belief is accepted and it contradicts the current observation. Intuitively, we might think that this could not happen. However, the defeasible structure of our rules and meta-rules permits such conclusions. After all, our current observations include the assertion that friend = no, even though we have the detection silhouette = friendly-shape. To resolve this situation we invoke a variant of the Persistence Rule:

(a) The Persistence Rule for Belief Selection.

When a previous belief about an attribute and a current observation of the same attribute are contradictory, and both values are supported by the current conditions, and one attribute value occurs *nb* times in *nb* decision cycles, then that value may be asserted as the current belief.

The variable *nb* is chosen to distinguish it from *n* in the original Persistence Rule. This is a user-settable value that will influence whether the agent places more emphasis on current observations as opposed to previous beliefs. Future research may reveal better solutions for this particular detail. However, other solutions that we considered involved many more rules and preconditions that had to be satisfied. In our implementation we focused on the real-time nature of the application and opted for a straightforward solution in order to avoid excessive rule-firing. In this particular example, the previous belief was not supported by the current conditions and we were able to assert the current observation as the new belief.

- (5) Transmit belief message to “decision-maker.” As with the previous example, this step is not part of the model per se. Including it permits us to make the same point we made earlier, that the belief message that is ultimately transmitted may contain attribute values that are inconsistent. On the other hand, the discrete-valued attributes may be adjusted to

match the value of the attribute *veh-name* if the particular implementation requires it. We opt to transmit the message “as is.”

This concludes our step-by-step discussion of two approaches to belief simulation: one involving a numerical method for handling uncertainty, and the other involving defeasible logic techniques for making belief selection. This preceding discussion was for the purpose of illustrating the mechanics of the two methods. We made no attempt to compare or evaluate the contrasting approaches. In the next section we present some results from simulations of the two methods and a discussion of the strengths and weaknesses that we were able to ascertain.

5.3. *Strengths and weaknesses of the two approaches*

We originally implemented the belief simulation model in a combat simulation system involving autonomous forces operating in a virtual environment portrayed on computer screens Branley, 1992. Consequently, we were able to collect data on the behaviour of our original model and on the overall performance of the agents receiving belief messages from the model. In presenting the research at hand, we focus our attention on comparing the behaviour of the two methods we have presented for selecting beliefs about discrete-valued attributes of objects. Therefore, we will present results pertaining strictly to that comparison and not to the overall performance of the agent.

Before discussing the simulations, we must state what we were looking for in terms of a performance measure and how we obtained that measure. Recall that the whole purpose of this model is to introduce imperfections into the beliefs of the autonomous agent. Our intent is that the degree of imperfection be governed, over time, by the value *m* which is a probabilistic measure of how accurately the agent is able to make observations under the given conditions. A frequentist interpretation of this probability is that if the agent were to make the observation many times, a fraction *m* of those observations would be accurate. This view enables us to design

the following kind of test for our model: we hold the input values constant and then, for a given number of trials, count the number of times that the belief values match the input values. We can say that a trial is *successful* if the output matches the input. If we divide the number of successes by the number of trials, we should, theoretically, obtain a number close to m .

There are two ways we can describe a trial as successful. We can say that the trial resulted in a *perfect match* if all of the input values match all of the output values (i.e., all of the object attributes match). We can say that a trial results in a *substantial match* if the attribute *veh-name* matches in both sets of values. This would mean that the object has been correctly identified, which might be all that is required in some implementations.⁴ Both interpretations of success are valid, so we will present both calculations in our discussions. Finally, the difference in the two matches (the perfect match will have a lower value, by definition) may be of interest since it indicates how often *veh-name* was identified correctly while other attributes were not; in other words it shows that there was a conflict between the values of all attributes.

Up to this point we have not considered an *overall m* value; we have computed m for each attribute; we obtained different numbers due to the effects of the distance factor, d . For the purposes of evaluating the models we must be clear about which m value we are using. When computing *substantial matches* as described above, we can use the m value for the attribute *veh-name* as an adequate performance measure. When evaluating *perfect matches* we must compute a special, overall, m value that reflects the probability of correctly identifying *all* of the object attributes. For now we will define an *overall m* as the average of the individual attribute m values.

Having presented our methodology, we can

Table 8

Input to the simulations: true attribute values of the target object

Attribute	True value
veh-name	T1
veh-type	tank
friend	yes
armament	main-gun
has-armor	has-armor
silhouette	enemy-shape
thermal-sig	enemy-sig

Table 9

Four of the five battlefield factors

Factor	Value
judgement	0.9
knowledge	0.8
visibility	1.0
equipment	0.5

now describe the simulations and the results. We ran both models through two versions of a simulation. In version 1, the agent is located at a point (0,0) and the target object is located at (2000,2000) in a two-dimensional plane. The distance between the two points is 2828. Table 8 shows the true attribute values of the target, and Table 9 shows the values for four of the factors that affect observation (see Factors). Recall that the distance factor may vary with each attribute. We held the target at a fixed location, thus keeping all variables constant, and ran both belief models through 200 trials. We ran two sets of trials under version 1: a) with the target at (2000,2000), and b) with the target held fixed further away from the agent at (5000,5000).

Table 10 depicts the results of this simulation. We see in Version 1a of the simulation that both models had a percentage of substantial (i.e., veh-

Table 10

Version 1 simulation results: target is at (2000, 2000)

Model	Overall m	Perfect matches	veh-name m	Substantial matches
Numerical	0.895	0.725	0.871	0.985
Logical	0.895	0.935	0.871	0.955

⁴ The choice of *veh-name* as this distinguished attribute is appropriate for the purpose of our test since *veh-name* may be directly observed, but is also inferred through various other attributes via decision rules.

Table 11

Version 1 simulation results: target is at (5000, 5000)

Model	Overall m	Perfect matches	veh-name m	Substantial matches
Numerical	0.858	0.495	0.858	0.975
Logical	0.858	0.715	0.858	0.765

name) matches that was above the m -value for veh-name. The values indicate that it is highly likely that the agent would have concluded (with qualifications) that the object under observation was a valid target. Depending on the nature of its interpretation processes, it might have destroyed the target object very early in the simulation. This may or may not be what the application developer intends. Next, note the difference between substantial matches and perfect matches. For the logical model, the difference is small; this corresponds to the logically consistent nature of the model. For the numerical model, however, the difference is very large. Certain attributes were identified incorrectly; the result of those misidentifications was to pull down the strength of belief in veh-name (since the value of these attributes were interpreted as additional evidence for veh-name) but not sufficiently enough to reverse the conclusion. This accounts for the large difference in perfect matches between the two models. In the logical case, the defeasible rule structure was able to overcome “noise” in the form of randomized error in the detection values in order to correctly deduce the exact identity of the object. In the numerical case, evidence values that supported opposing conclusions had a “voice” due to the evidence combination process. Then, randomization at the *end* of the process did not always work in favour of the true values.

Having noted these facts, consider what happens when we run the same simulation at a greater distance, version 1b. In Table 11, the target object is at P(5000,5000) while the agent remains at P(0,0).

Now we see the effects of increased range on the belief simulation process. In the numerical case, evidence combination produces a consistently high percentage of substantial matches, which would lead to the agent destroying the

target quickly (depending on how its interpretative processes operate). This is because even a small evidence value can influence the process. However, because of noise in the sensory readings, due to greater distance, the numerical model's percentage of perfect matches dropped significantly. In the logical case, the percentage of both perfect and substantial matches dropped far below their previous readings, but remained relatively close together. The difference between substantial matches for the two models was as before, only a bit more pronounced for the numerical model. After studying the results of many simulations, we notice that the logical model appears to be more internally consistent and more stable, rising and falling rather consistently as m rises and falls.

In the second simulation we used the same attribute values for the target, but gave the target an initial location of P(4000,4000) and decremented the x and y coordinates by a value of 200 at regular intervals, thus moving the target in a straight line toward the agent, which was fixed at P(0,0). Four of the observation factors were held to the same values as before, but the fifth, dis-

Table 12

Distances and m values for the moving target simulation

Cycle	Distance	Overall m	Veh-name m
0	5657	0.858	0.860
1	5374	0.858	0.861
2	5091	0.858	0.862
3	4808	0.858	0.864
4	4525	0.858	0.866
5	4243	0.858	0.868
6	3960	0.858	0.871
7	3677	0.859	0.875
8	3394	0.861	0.880
9	3111	0.865	0.886
10	2828	0.871	0.895
11	2546	0.879	0.905
12	2263	0.889	0.916
13	1980	0.902	0.931
14	1697	0.917	0.947
15	1414	0.935	0.967
16	1131	0.958	0.985
17	849	0.984	0.989
18	566	0.990	0.990
19	283	0.990	0.990

Table 13
Version 2 simulation statistics: moving target

Model	Overall m (average)	Perfect matches	veh-name m (average)	Substantial matches
Numerical	0.91	0.7	0.895	0.85
Logical	0.91	0.9	0.895	0.9

tance, changed as the target moved closer to the agent, thus the m values changed. At each “stopping point,” or decision cycle, we calculated a set of agent beliefs. We did this 20 times to show, on average, how the models reacted to moving targets. This corresponds to the way in which these models would be used in practice, except that the target object would move in much smaller increments and beliefs would be computed more often. In Table 12 we have provided the actual distance figures and m values for this simulation, as well as the summarized information in Table 13.

As in Version 1 of the simulation, we see in Version 2 the trend that the perfect and substantial matches in the logical case are close together (in this case, the same) while those same percentages for the numerical model are farther apart. This provides additional support for our observation that the logical model appears to be more stable and it also appears to rise and fall consistently with m .

6. Conclusion

We have described a general framework, and presented two methods, for implementing an autonomous agent belief simulation system. Our framework for agent reasoning explicitly separates the agent’s decision-making processes from the information-gathering ones. In the absence of any sensors to gather information, we introduce a probabilistic observation model which acts as the agent’s interface to collect information from the outside world. We combine these external observations with what the agent already knows, or believes, in order to form the agent’s new beliefs. In doing so, the agent is often faced with multiple, and often conflicting, pieces of related evi-

dence. For reasoning with these evidences, we presented a numerical approach, based upon the Dempster-Shafer evidence combination algorithm, and a logical approach, based upon defeasible reasoning. We presented details of the implementation of the two approaches and then discussed results of several simulations that we conducted in order to evaluate the models.

Our conclusion thus far in this research is that the logical approach has certain behavioral characteristics that make it more stable, and thus more predictable. As we pointed out in the previous section, the frequency with which the agent had correct information using the logical approach tended to rise and fall with the indicator, m . This is the desired behaviour. This result is partly due to where the randomization is inserted. In the logical method it is applied at the beginning of a decision cycle, after which the agent’s rules have a chance to reason correctly in spite of noisy input. In the numerical approach, randomization is used in the selection of weighted options, which leaves open the possibility of incorrect beliefs being chosen in spite of evidence to the contrary. Which method is more desirable depends, to a large extent, on the reasoning capabilities of the autonomous agent for whom the beliefs are being generated. For example, if the agent could reason with belief *intervals*, the Dempster Shafer method (sans the step in which we randomly generated a point belief from the interval) would be desirable.

Many more issues can be addressed in this area. For example, we have not addressed how the agent’s beliefs are influenced by other agents. Nor have we addressed the important role that perception undoubtedly plays in the formation of beliefs. In addition to these research directions, there are many potential applications for this kind of model. Besides the case of using a simulator to train soldiers, one could conduct a simulation of a conflict by having two teams, or forces, of autonomous agents oppose each other. In this type of study, it would be essential for the agents to be acting upon beliefs that are influenced by environmental conditions and each agent’s own (simulated) capabilities. In simulations of other settings, such as economies and societies, groups

of agents that interact freely, but which are constrained by imperfect beliefs, may greatly enhance the decision-making value of the simulation.

References

- Branley, W.C., Modelling Observation in Intelligent Agents: Knowledge and Belief. Master's thesis, Naval Postgraduate School, Monterey, CA (1992).
- Brewka, G., Nonmonotonic Reasoning: Logical Foundations of Commonsense, Cambridge University Press (1991).
- Ceranowicz, A., ModSAF Capabilities, Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation, Institute for Simulation and Training, Orlando, FL (1994a). Ceranowicz, A., Operator Control of Behaviour in ModSAF, Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation, Institute for Simulation and Training, Orlando, FL (1994b). Davis, E., Representations of Commonsense Knowledge, Morgan Kaufmann Publishers, Inc. (1990).
- Genesereth, M.R., and N.J. Nilsson, Logical Foundations of Artificial Intelligence, Morgan Kaufmann (1987).
- Gordon, J., and E.H. Shortliffe, The Dempster-Shafer Theory of Evidence, in B.G. Buchanan and E.H. Shortliffe (eds.), Rule-based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project, Addison-Wesley Publishing Company, Reading, MA (1984) pp. 272–292.
- Homer, The Iliad, 750 B.C., translation by Samuel Butler, Washington Square Press, New York (1969). Konolige, K., A Deduction Model of Belief, Morgan Kaufmann Publishers, Inc., Los Altos, CA (1986).
- Nute, D., Defeasible Logic and the Frame Problem, in Knowledge Representation and Defeasible Reasoning, Studies in Cognitive Systems, Kluwer Academic Publishers, Boston (1990).
- Nute, D., Basic defeasible logic, in Intensional Logics for Programming, Del Cerro, L.F., and Penttonen, M., editors, Clarendon Press, Oxford (1992).
- Pearl, J., Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers, Inc., San Mateo, CA (1988). Pope, A., The SIMNET Network and Protocols, BBN Report No. 7102, BBN Systems and Technologies, Cambridge, MA (July 1989).
- Pratt, D.R., Bhargava, H.K., Culpepper, M., Locke, J., Collaborative Autonomous Agents in the NPSNET Virtual World, Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation, Institute for Simulation and Training, Orlando, FL (1994). Robinson, C.A., Jr., Simulation Improves Desert Combat Operations, SIGNAL, Armed Forces Communications and Electronics Association (July 1991).
- Roehrig, Stephen, Probabilistic Inference and Path Analysis, Decision Support Systems (forthcoming 1995).
- Rosenbloom, P.S., Laird, J.E., Newell, A., The SOAR Papers: Research on Integrated Intelligence, MIT Press, Cambridge, MA (1993).
- Rosenbloom, P.S., Johnson, W.L., Jones, R.M., Koss, F., Laird, J.E., Lehman, J.F., Rubinoff, R., Schwamb, K.B., Tambe, M., Intelligent Automated Agents for Tactical Simulation: A Progress Report, Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation, Institute for Simulation and Training, Orlando, FL (1994).
- Sanchez, E., and L.A. Zadeh (ed.), Approximate Reasoning in Intelligent Systems, Decision, and Control, Pergamon Press, New York, NY (1987).
- Shafer, G., A Mathematical Theory of Evidence, Princeton University Press, Princeton, NJ (1976). Shafer, G., and J. Pearl (ed.), Readings in Uncertain Reasoning, Morgan Kaufmann Publishers, San Mateo, CA (1990).
- Merriam-Webster Inc., Webster's Ninth New Collegiate Dictionary, Merriam-Webster Inc., Springfield, MA (1989).
- Zyda, M.J., and D.R. Pratt, NPSNET: A 3D Visual Simulator for Virtual Exploration and Experimentation, Digest of Technical Papers XXII, SID International Symposium, 8 (May 1991).
- Zyda, M.J., D.R. Pratt, J.G. Monahan, and K.P. Wilson, NPSNET: Constructing a 3D Virtual World, Computer Graphics Special Issue on the 1992 Symposium on Interactive 3D Graphics, March 29–April 1 (1992).